Amendments to the Specification

Please replace paragraphs [0002]-[0005], [0007]-[0008], [0010]-[0012], [0078]-[0082], [0084]-[0085], [0087]-[0093], [0096], [0098]-[0099], [0102]-[0103], [0110]-[0111], [0115]-[0130], [0180]-[0181], [0197], [0229], and [0332] with following paragraphs as amended.

[0002] Japanese Published Unexamined Patent Application No. 11-327908 discloses an agent server. It discloses that the agent server limits the number of active agents in order to ~~curb~~ control increase in loads and notifies each ~~user~~ member of a predetermined message by using an agent. It is incorporated herein by reference in entirety for all purposes.

[0003] The following describes problems that are to be solved by the present invention. As for a concrete example of the message processing server, there is a mail delivery server for notifying a member of acceptance of predetermined cause information. Concrete needs of such a mail delivery server will be considered here. For instance, as an expected need, the member pre-registers with a subscription table a condition for giving a notice desired by the member (ex. a notice that the IBM's stock price became P yen or higher). And if the message processing server receives ~~a~~ a message such as fluctuations in stock prices from a client (ex. a computer provided to a predetermined securities company for receiving stock prices of enterprises online), it identifies the member whose cause is the ~~a~~ message from the subscription table by performing a search and gives a notice to the identified member.

[0004] An agent server of the Japanese Published Unexamined Patent Application No. 11-327908 ~~patent document 1 and the message processing server of the non-patent document 1~~ neither pre-registers with the subscription table the condition for giving a notice to each member to be notified nor searches for the member to be notified ~~this time~~ as to the ~~a~~ message from the subscription table and delivers notice mail to the member.

[0005] The following need is also expected as to the mail delivery server. ~~To be more specific, in~~ In the case where different ~~a~~s messages arise closely ~~time-wise~~ and the notice

~~mail~~ mails relating to ~~each of~~ the different ~~x1s~~ messages ~~is~~ are delivered to the same member, it is desired to assign ~~a priority~~ priorities to ~~each of~~ the notice ~~mail~~ mails ~~so as~~ to send the ~~messages~~ notice mails in order of ~~priority~~ priorities. For instance, a typical member to be notified wishes to receive the notice mail relating to stock price information earlier than the notice mail of other contents such as new product introduction in a predetermined field. A notice mail message processing application implemented on the server utilizes a multi-thread instead of a single thread for the sake of reducing overall processing time. Therefore, if the thread is allocated to each ~~x1~~ message and the message is inserted into a message queue related to each member, it takes longer time to insert the message having a large number of total members into the message queue for each member than to insert the message having a small number of total members therein so that of a high priority may get to the member later than that of a low priority.

[0007] As another expected need, there will be the cases where, when there are a plurality of pieces of notice mail of different contents to be delivered to the same member, they must be delivered in order of acceptance of the cause information thereof. For instance, in the case where the IBM's stock price becomes P1 and then becomes P2 (P2<P1) several seconds later, the member to be notified as to the notice mail caused by both the ~~x1s~~ messages must receive the notice mail related to the P1 first and then receive the notice mail related to the P2. Otherwise, the member will misunderstand it as a rise in the stock price although it is a drop in the stock price. In the case where a multi-thread message processing application is developed, the application has the processing performed by a plurality of threads in parallel. Therefore, depending on size of a throughput of each thread, the thread allocated to the later ~~x1~~ message finishes the processing earlier than the thread allocated to the earlier ~~x1~~ message so that the notice mail related to the ~~x1~~ message later time-wise may get to the member to be notified earlier than the notice mail related to the ~~x1~~ message earlier time-wise.

[0008] Thus, an aspect of the present invention is to reduce work time in a message processing apparatus, a message processing method and a message processing program for~~, as~~

~~to~~ the message applied to a process requester corresponding to an ~~agent start cause event~~ agent activating event, causing a corresponding agent to perform a predetermined process.

[0010] A further aspect of the present invention is to provide the message processing apparatus, message processing method and message processing program for ensuring that, while ~~realizing the processing with the multi-thread~~ performing multi-thread processing, the messages applied to the process requestors are processed in order of acceptance of the ~~agent start cause events~~ agent activating events which caused them.

[0011] The message processing apparatus according to the present invention has: process requestor search information management means for managing process requestor search information for searching for an applicable process requestor as to an ~~agent start cause event~~ agent activating event; acceptance means for accepting the ~~agent start cause event~~ agent activating event; list information creation means for, based on the above described process requestor search information, creating list information on process requesters to which a message generated ~~due to~~ upon the above described ~~agent start cause event~~ agent activating event ~~is applied~~; a plurality of agents associated with the process requestors, stored in a persistent storage, readable from the persistent storage to a cache memory as which is a program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in a message queue corresponding to the agent; insertion and reading means for, of the process requestors included in the above described list information, selecting a plurality of unselected ones as the process requestors to be inserted and read, inserting the above described message into the message queues related to the process requesters to be inserted and read and reading the agents related to the above described process requestors from the persistent storage to the cache memory; agent instruction means for instructing the agent related to the message queue having the message inserted therein to operate; and repetitive instruction means for, in the case where the unselected one remains among the process requestors included in the above described list information, waiting for termination of the process of all the agents in operation and instructing the above described insertion and reading means to repeat the process.

[0012] An example of a message processing apparatus according to the present invention has: the process requester search information management means for managing the process requestor search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance means for accepting the ~~agent start cause event~~ agent activating event; the list information creation means for, based on the above described process requestor search information, creating the list information on the process requestors to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied; the plurality of agents associated with the process requesters, stored in the persistent storage, readable from the persistent storage to the cache memory ~~as~~ which is the program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; a first message queue processing mechanism; a second message queue processing mechanism; selection means for selecting either one of the first and second message queue processing mechanisms; and the agent instruction means for, as to the agents related to the message queue having the message inserted therein, immediately instructing the agent to operate if the agent is in the cache memory, and reading the agent from the above described persistent storage to the above described cache memory and then instructing the agent to operate if the agent is not in the cache memory.

[0078] The present invention provides methods, systems and apparatus to reduce work time in a message processing apparatus, a message processing method and a message processing program for, as to the message applied to a process requestor corresponding to an ~~agent start cause event~~ agent activating event, causing a corresponding agent to perform a predetermined process.

[0079] The present invention also provides a message processing apparatus, message processing method and message processing program for controlling reading of an agent from a persistent storage to a cache memory and achieve the processing based on the priorities

while reducing the overall processing time. The present invention further provides a message processing apparatus, message processing method and message processing program for ensuring that, while realizing the processing with the multi-thread, the messages applied to the process requestors are processed in order of acceptance of the ~~agent start cause events~~ agent activating events which caused them.

[0080] A message processing apparatus according to the present invention has: process requestor search information management means for managing process requestor search information for searching for an applicable process requestor as to an ~~agent start cause event~~ agent activating event; acceptance means for accepting the ~~agent start cause event~~ agent activating event; list information creation means for, based on the above described process requestor search information, creating list information on process requesters to which a message generated ~~due to~~ upon the above described ~~agent start cause event~~ agent activating event ~~is applied~~; a plurality of agents associated with the process requesters, stored in a persistent storage, readable from the persistent storage to a cache memory ~~as~~ which is a program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in a message queue corresponding to the agent; insertion and reading means for, of the process requestors included in the above described list information, selecting a plurality of unselected ones as the process requesters to be inserted and read, inserting the above described message into the message queues related to the process requestors to be inserted and read and reading the agents related to the above described process requestors from the persistent storage to the cache memory; agent instruction means for instructing the agent related to the message queue having the message inserted therein to operate; and repetitive instruction means for, in the case where the unselected one remains among the process requesters included in the above described list information, waiting for termination of the process of all the agents in operation and instructing the above described insertion and reading means to repeat the process.

[0081] To operate the agents, it is necessary for the agents to exist in the cache memory. The process requesters, that is, the agents may reach several hundred thousand to several million

or more for instance, and so it is difficult to have all the agents constantly existing in the cache memory in terms of a capacity of the cache memory. According to the present invention, the list information on the process requestors to which the message generated due to the ~~agent start cause event~~ agent activating event is applied is created based on the process requestor search information, and all (when there are a small number of unselected process requestors) or several (when there are a large number of unselected process requesters) unselected process requestors remaining in the list information are selected so as to insert the message into the message queues associated with the selected process requestors. The insertion and reading means reads the agents associated with the process requesters from the persistent storage to the cache memory in parallel with or before or after the insertion work. Thus, as for the message queues having the message inserted therein, the agents associated with the message queues start operating according to an instruction from the agent instruction means such as a scheduler, where the agents to be operated have their existence in the cache memory ensured so that the reading of the agents from the persistent storage to the cache memory is controlled and the work time is reduced.

[0082] The ~~agent start cause events~~ agent activating events include a change in a stock price, a price change of a product database in a cybermall system and so on, for instance. The agents are not limited to those of the same kind of start causes, such as the ~~agent start cause events~~ agent activating events related to the IBM's stock price. For instance, the same agent may have a plurality of kinds of ~~agent start cause events~~ agent activating events as the start causes, such as having as the start causes the ~~agent start cause event~~ agent activating event as an update of a database related to prices of personal computers and the ~~agent start cause event~~ agent activating event as an update of a database related to prices of printers. The message processing apparatus according to the present invention is not limited to a mail delivery processing apparatus. The same agent may notify the process requester of the mail as to a certain ~~agent start cause event~~ agent activating event (the ~~agent start cause event~~ agent activating event is referred to as an ~~agent start cause event~~ agent activating event A1), but may not notify the process requestor of the mail as to another certain ~~agent start cause event~~ agent activating event (the ~~agent start cause event~~ agent activating event is referred to as an

~~agent start cause event~~ agent activating event A2). In addition, as to the same ~~agent start cause event~~ agent activating event (the ~~agent start cause event~~ agent activating event is referred to as an ~~agent start cause event~~ agent activating event A3), the ~~agent start cause event~~ agent activating event may have different processing results, that is, it may or may not notify the corresponding process requestor of the mail for instance, depending on a difference in one or a plurality of ~~agent start cause events~~ agent activating events which became the start causes before occurrence of the ~~agent start cause event~~ agent activating event A3 and the difference in their combinations. Moreover, the case of applying the message processing apparatus according to the present invention to an apparatus other than the mail delivery apparatus will be described in detail in a fourth embodiment described later.

[0084] The agent typically includes a message handler for performing a predetermined process by using the message in the message queue as an argument and the data used by the message handler. For instance, if the message generated due to the ~~agent start cause event~~ agent activating event A is a message B, the message B is inserted into all the message queues associated with all the process requesters to which the message is applied, where each process requestor may have different processing result in the agent as to the message B according to the data in the agent. For instance, when the stock price of an enterprise 1 becomes 100 dollars from 90 dollars, process requestors 1 and 2 receive the notice mail indicating that it became 100 dollars or more, and a process requestor 3 receives the notice mail indicating that it became 95 dollars or more.

[0085] Another example of a message processing apparatus according to the present invention has: the process requester search information management means for managing the process requestor search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance means for accepting the ~~agent start cause event~~ agent activating event; the list information creation means for, based on the above described process requestor search information, creating the list information on the process requestors to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied; the plurality of agents associated with the process

requestors, stored in the persistent storage, readable from the persistent storage to the cache memory ~~as~~ which is the program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; a first message queue processing mechanism; a second message queue processing mechanism; selection means for selecting either one of the first and second message queue processing mechanisms; and the agent instruction means for, as to the agents related to the message queue having the message inserted therein, immediately instructing the agent to operate if the agent is in the cache memory, and reading the agent from the above described persistent storage to the above described cache memory and then instructing the agent to operate if the agent is not in the cache memory.

[0087] The total number of the process requesters related to the ~~agent start cause event~~ agent activating event is different as to each ~~agent start cause event~~ agent activating event. It may be a very large number or an extremely small number. The rate at which the agents associated with the process requestors are hit in the cache memory is different according to the situation. When similar ~~agent start cause events~~ agent activating events are successively accepted, the possibility that the process requestors related to each ~~agent start cause event~~ agent activating event coincide becomes higher, and so a hit rate in the cache memory rises as to the agents associated with the process requestors to which the message generated ~~due to~~ upon the ~~agent start cause event~~ agent activating event ~~is applied~~. In such cases, the work time may sometimes be reduced, (a) rather than by, of the process requesters included in the list information, selecting the unselected ones, inserting the message into all the message queues associated with the selected process requesters and reading the agents related to all the message queues to the cache memory, (b) by immediately reading the agent to be operated when the agent is in the cache memory or reading the agent from the persistent storage to the cache memory and then operating it when the agent is not therein. According to the present invention, (a) and (b) are freely selectable. (a) and (b) are switchable for each ~~agent start cause event~~ agent activating event, each day of the week, each season and each time period for instance.

[0088] A further message processing apparatus according to the present invention has: the process requestor search information management means for managing the process requestor search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance means for accepting the ~~agent start cause event~~ agent activating event; the process requestor determination means for determining the process requestor to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied based on the above described process requestor search information; the plurality of agents associated with the process requestors, stored in the persistent storage, readable from the persistent storage to the cache memory as the program execution area and abandonable from the cache memory, each agent becoming operable when existing in the cache memory; at least one sub-process priority determination means for determining process priority about each message as sub-process priority based on a single standard of value; compound process priority determination means for, when the total number of the above described sub-process priority determination means is two or more, determining compound process priority about each message based on the sub-process priority individually determined as to each message by each sub-process priority determination means, and when the total number of the above described sub-process priority determination means is one, determining as the compound process priority the sub-process priority determined by the above described one sub-process priority determination means as to each message; and agent instruction means for rendering the message of the highest compound process priority among the messages held by each message queue as the message of the highest priority, and between the agents related to the message queues of which compound process priority of the message of the highest priority is the same, instructing the agent existing in the cache memory to operate in preference to the agent not existing therein.

[0089] There is a need for, while controlling the processing time of the entire process requestor processing apparatus, setting processing priorities as to the messages and processing the messages based on the processing priorities. The processing priorities may be determined based on a single standard of value or determined by compounding the processing priorities of

a plurality of standards of value. According to the present invention, the processing is performed based on compound processing priorities, and in the case where the messages of the same highest compound processing priority are included among the message queues, the message queue having the agent associated with the message queue existing in the cache memory is processed in preference to the message queue not having it. Thus, it eliminates the situation in which the agent to be operated as to the ~~agent start cause event~~ agent activating event of this time is not operated while existing in the cache memory and its operation is put off so that its turn of delivery comes when it no longer exists in the cache memory, and the agent has to be read from the persistent storage to the cache memory.

[0090] A further message processing apparatus according to the present invention has: the process requestor search information management means for managing the process requester search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance means for accepting the ~~agent start cause event~~ agent activating event; acceptance order information management means for managing the acceptance order information on the ~~agent start cause events~~ agent activating events accepted by the above described acceptance means; the plurality of agents associated with the process requestors, stored in the persistent storage, readable from the persistent storage to the cache memory as the program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; a plurality of threads mutually operable in parallel, each thread detecting the process requesters to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied based on the above described process requestor search information and inserting the above described message into the message queues related to the process requesters; allocation means for allocating to each thread the ~~agent start cause event~~ agent activating event to be processed by that thread; proceeding information management means for managing proceeding information on the process by the thread as to each ~~agent start cause event~~ agent activating event accepted by the above described acceptance means; determination means for, as to the ~~agent start cause event~~ agent activating event of which process proceeding information is the information on

thread process termination (hereafter, referred to as a "determined ~~agent start cause event~~ agent activating event "), determining whether or not, of the ~~agent start cause events~~ agent activating events accepted by the above described acceptance means prior to the determined ~~agent start cause event~~ agent activating event, there is any ~~agent start cause event~~ agent activating event of which thread process is unfinished; and agent control means for controlling the process by the agent as to the message generated due to the determined ~~agent start cause event~~ agent activating event determined as "yes" by the above described determination means.

[0091] The ~~agent start cause events~~ agent activating events are accepted one after another, and the number of the process requesters to which the message generated due to the ~~agent start cause event~~ agent activating event is applied is enormous. Therefore, for the sake of reducing the work time, it is desirable, as to each ~~agent start cause event~~ agent activating event, to have different threads, that is, a plurality of threads perform the process of inserting the message generated due to the ~~agent start cause event~~ agent activating event into the message queue of the process requestor related to the ~~agent start cause event~~ agent activating event. In the case of adopting a multi-thread, however, the thread for processing the ~~agent start cause event~~ agent activating event accepted later in acceptance order may finish the process earlier than the thread for processing the ~~agent start cause event~~ agent activating event accepted earlier in acceptance order depending on the total number of the process requestors to which the message generated due to each ~~agent start cause event~~ agent activating event is applied. According to the present invention, even if the process related to the ~~agent start cause event~~ agent activating event accepted later in acceptance order is finished in the process of inserting the messages into the message queues, the process to the process requesters by the agents is controlled as to the message generated due to the later ~~agent start cause event~~ agent activating event unless the process related to the ~~agent start cause event~~ agent activating event accepted earlier in acceptance order is finished.

[0092] The message processing method according to the present invention has: a process requestor search information management step of managing process requestor search

information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; an acceptance step of accepting the ~~agent start cause event~~ agent activating event; a list information creation step of, based on the above described process requestor search information, creating the list information on the process requesters to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied; an agent setting step of setting a plurality of agents, the agents being associated with the process requesters, stored in the persistent storage, readable from the persistent storage to the cache memory as the program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; an insertion and reading step of, of the process requesters included in the above described list information, selecting a plurality of unselected ones as the process requesters to be inserted and read, inserting the above described message into the message queues related to the process requestors to be inserted and read, and reading the agents related to the above described process requestors from the persistent storage to the cache memory; an agent instruction step of instructing the agent related to the message queue having the message inserted therein to operate; and a repetitive instruction step of, in the case where the unselected one remains among the process requesters included in the above described list information, waiting for termination of the process of all the agents in operation and instructing the above described insertion and reading step to be repeated.

[0093] Another message processing method according to the present invention has: the process requester search information management step of managing the process requestor search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance step of accepting the ~~agent start cause event~~ agent activating event; the list information creation step of, based on the above described process requestor search information, creating the list information on the process requestors to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied; the agent setting step of setting a plurality of agents, the agents being associated with the process requestors, stored in the persistent storage, readable

from the persistent storage to the cache memory as the program execution area and abandonable from the cache memory, each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; a first message queue processing step; a second message queue processing step; a selection step of selecting either one of the first and second message queue processing steps; and

[0096] A further message processing method according to the present invention has: the process requestor search information management step of managing the process requestor search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance step of accepting the ~~agent start cause event~~ agent activating event; a process requester determination step of determining the process requestor to which the message generated due to the above described ~~agent start cause event~~ agent activating event is applied based on the above described process requestor search information; the agent setting step of setting a plurality of agents, the agents being associated with the process requestors, stored in the persistent storage, each agent being readable from the persistent storage to the cache memory as the program execution area and abandonable from the cache memory, and each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; at least one sub-process priority determination step of determining a process priority about each message as a sub-process priority based on a single standard of value; a compound process priority determination step of, when the total number of the above described sub-process priority determination steps is two or more, determining a compound process priority about each message based on the sub-process priority individually determined as to each message in each sub-process priority determination step, and when the total number of the above described sub-process priority determination steps is one, determining as the compound process priority the sub-process priority determined in the above described one sub-process priority determination step as to each message; and

[0098] A still further message processing method according to the present invention has: the process requester search information management step of managing the process requestor

search information for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event; the acceptance step of accepting the ~~agent start cause event~~ agent activating event; an acceptance order information management step of managing the acceptance order information on the ~~agent start cause events~~ agent activating events accepted by the above described acceptance step; the agent setting step of setting a plurality of agents, the agents being associated with the process requestors, stored in the persistent storage, readable from the persistent storage to the cache memory as the program execution area and abandonable from the cache memory, and each agent operating only when existing in the cache memory to be able to process the message in the message queue corresponding to the agent; a thread setting step of setting a plurality of threads, the threads being mutually operable in parallel, detecting the process requesters to which the message generated due to the ~~agent start cause event~~ agent activating event is applied based on the above described process requester search information and inserting the above described message into the message queues related to the process requesters; an allocation step of allocating to each thread the ~~agent start cause event~~ agent activating event to be processed by that thread; a proceeding information management step of managing the proceeding information on the process by each thread as to each ~~agent start cause event~~ agent activating event accepted by the above described acceptance step; a determination step of, as to the ~~agent start cause event~~ agent activating event of which process proceeding information is the information on thread process termination (hereafter, referred to as the "determined ~~agent start cause event~~ agent activating event "), determining whether or not, of the ~~agent start cause events~~ agent activating events accepted in the above described acceptance step prior to the determined ~~agent start cause event~~ agent activating event, there is any ~~agent start cause event~~ agent activating event of which thread process is unfinished; and

[0099] an agent control step of controlling the process by the agent as to the message generated due to the determined ~~agent start cause event~~ agent activating event determined as "yes" in the above described determination step.

[0102] FIG. 1 is a schematic view of a message processing system 10. A network 12 is the Internet in the typical cases. A message processing server 14 for implementing the applications following the present invention, a plurality of information source clients 15, and several hundred thousand to several million or more process requestor computers 16 can mutually send and receive messages, data and various kinds of information via the network 12. In the case where a predetermined portion in a work flow is the process requestor, it is possible for one computer 16 to be a plurality of mutually identifiable process requestors. The information source clients 15 are installed in securities companies and so on in order to send stock quotations as an ~~agent start cause event~~ agent activating event to the message processing server 14, for instance. The requestor computers 16 may be connected to the network 12 directly or via a router.

[0103] FIG. 2 is a block diagram of a message processing apparatus 20. In the message processing apparatus 20, process requestor search information management means 22 manages process requestor search information 21 for searching for an applicable process requestor as to an ~~agent start cause event~~ agent activating event. The process requester search information 21 is stored in a persistent storage such as a hard disk drive and a magnetic tape drive. Acceptance means 27 accepts the ~~agent start cause event~~ agent activating event. List information creation means 28 creates list information on process requesters to which a message generated due to the ~~agent start cause event~~ agent activating event is applied based on the process requestor search information 21. A plurality of agents 23 are associated with the process requestors, stored in a persistent storage 24, readable from the persistent storage 24 to a cache memory 25 as a program execution area and abandonable from the cache memory 25. Each agent 23 operates only when existing in the cache memory 25 to be able to process the message in a message queue corresponding to the agent 23. Insertion and reading means 38 selects a plurality of unselected ones as the process requestors to be inserted and read of the process requestors included in the list information, inserts the message into a message queue 39 related to the process requestors to be inserted and read and reads the agents 23 related to the process requestors from the persistent storage 24 to the cache memory 25. Agent instruction means 31 instructs the agents 23 related to the message queue 39 having

the message inserted therein to operate. Repetitive instruction means 32 waits for termination of the process of all the agents 23 in operation and instructs the insertion and reading means 38 to repeat the process in the case where the unselected one remains among the process requesters included in the list information.

[0110] With reference to FIG. 4 again, the selection by the selection means 43 is based on an instruction of an operator. Or the selection by the selection means 43 is based on the estimated number of the process requestors to which the message generated due to the ~~agent start cause event~~ agent activating event of this time is applied, estimated hit rate in the cache memory 25 as to the agents 23 related to the process requesters to which the message generated due to the ~~agent start cause event~~ agent activating event of this time is applied, estimated work time, in the case where the selection means 43 selects the first message queue processing mechanism 44, from acceptance of the information by the acceptance means 27 until obtaining the list information on the process requestors to which the message is applied and inserting the message into the message queues 39 of all the agents 23, estimated work time, in the case where the selection means 43 selects the second message queue processing mechanism 45, from acceptance of the information by the acceptance means 27 until obtaining the list information on the process requesters to which the message is applied and inserting the message into the message queues 39 of all the agents 23, estimated time from determination of use as to the agent 23 in the cache memory 25 until completion of the process by the determined agent 23, and/or estimated time from the determination of use as to the agent 23 outside the cache memory 25 until the completion of the process by the determined agent 23.

[0111] FIG. 7 is a block diagram of a message processing apparatus 52. The process requestor search information 21, process requestor search information management means 22, agents 23, persistent storage 24, cache memory 25, acceptance means 27 and message queue 39 in the message processing apparatus 52 are the same as those in the aforementioned message processing apparatus 20, and a description thereof will be omitted. Process requestor determination means 53 determines the process requestor to which the message generated due to the ~~agent start cause event~~ agent activating event is applied based on the process requestor

search information 21. At least one sub-process priority determination means 54a, 54b, . . . determine a process priority about each message as sub-process priority based on a single standard of value. Compound process priority determination means 55 determines a compound process priority about each message based on the sub-process priority individually determined as to each message by each sub-process priority determination means 54a, 54b, . . . when the total number of the sub-process priority determination means 54a, 54b, . . . is two or more. And when the total number of the sub-process priority determination means is one, the compound process priority determination means 55 determines as the compound process priority the sub-process priority determined by the one sub-process priority determination means (54a for instance) as to each message. Agent instruction means 56 renders the message of the highest compound process priority among the messages held by each message queue as the message of the highest priority, and instructs the agent 23 existing in the cache memory 25 to operate in preference to the agent 23 not existing therein between the agents 23 related to the message queues of which compound process priority of the message of the highest priority is the same.

[0115] FIG. 9 is a block diagram of a message processing apparatus 64. The process requestor search information 21, process requestor search information management means 22, agents 23, persistent storage 24, cache memory 25, acceptance means 27 and message queue 39 in the message processing apparatus 64 are the same as those in the aforementioned message processing apparatus 20, and a description thereof will be omitted in order to mainly describe differences. Acceptance order information management means 65 manages acceptance order information on the ~~agent start cause events~~ agent activating events accepted by the acceptance means 27. A plurality of threads 67a, 67b . . . are mutually operable in parallel. The threads 67a, 67b . . . detect the process requesters to which the message generated due to the ~~agent start cause event~~ agent activating event is applied based on the process requestor search information 21 and inserts the message into the message queues 39 related to the process requestors. Allocation means 66 allocates to each of the threads 67a, 67b . . . the ~~agent start cause event~~ agent activating event to be processed by that thread. Proceeding information management means 70 manages proceeding information on the process by the threads 67a,

67b . . . as to each ~~agent start cause event~~ agent activating event accepted by the acceptance means 27. As for the ~~agent start cause event~~ agent activating event of which process proceeding information is the information on process termination of the threads 67a, 67b . . . (hereafter, referred to as a "determined ~~agent start cause event~~ agent activating event"), determination means 72 determines whether or not, of the ~~agent start cause events~~ agent activating events accepted by the acceptance means 27 prior to the determined ~~agent start cause event~~ agent activating event, there is any ~~agent start cause event~~ agent activating event of which processing by the threads 67a, 67b . . . is unfinished. Agent control means 73 controls the process by the agent 23 as to the message generated due to the determined ~~agent start cause event~~ agent activating event when it is determined ~~as "yes"~~ that there is any agent activating event of which processing by threads is unfinished by the determination means 72. The agent control means 73 allows the process by the agent 23 as to the message generated due to the determined ~~agent start cause event~~ agent activating event when it is determined ~~as "no"~~ that there is no agent activating event of which processing by the threads is unfinished by the determination means 72.

[0116] In the case where the ~~agent start cause event~~ agent activating event immediately following the ~~agent start cause event~~ agent activating event determined as "no" in acceptance order is already determined as ~~"yes,"~~ that there is any agent activating event of which processing by threads is unfinished, the determination means 72 changes the determination result from ~~"yes"~~ that there is any agent activating event of which processing by threads is unfinished to ~~"no."~~ that there is no agent activating event of which processing by the threads is unfinished The agents 23 in FIG. 9 are designed to continuously process the plurality of continuous messages in the case of processing the message queue 39 in which the messages generated due to a plurality of ~~agent start cause events~~ agent activating events of which determination result by the determination means 72 is ~~"no" continue in an acceptance order direction~~ that there is no agent activating event of which processing by the threads is unfinished.

[0117] FIG. 10 is a flowchart of a message processing method. In S100 (process requester

search information management step), the process requestor search information 21 for searching for the applicable process requestor as to the ~~agent start cause event~~ agent activating event is managed. In S1104 (acceptance step), the ~~agent start cause event~~ agent activating event is accepted. In S105 (list information creation step), the list information on the process requesters to which the message generated due to the ~~agent start cause event~~ agent activating event is applied is created based on the process requestor search information 21. In S101 (agent setting step), a plurality of agents 23 are set up. In this setup, the agents 23 are associated with the process requestors, stored in the persistent storage 24, readable from the persistent storage 24 to the cache memory 25 as the program execution area and abandonable from the cache memory 25, and each agent 23 operates only when existing in the cache memory 25 to be able to process the message in the message queue corresponding to the agent 23. S106 (insertion and reading step) will be described later by referring to FIG. 11. In S108 (agent instruction step), the agent 23 related to the message queue 39 having the message inserted therein is instructed to operate. In S107 (repetitive instruction step), in the case where the unselected one remains among the process requesters included in the list information, the termination of the process of all the agents 23 in operation is awaited and an instruction to repeat S106 (insertion and reading step) is provided.

[0118] FIG. 11 is a detailed view of S1106 (insertion and reading step) in the flowchart in FIG. 10. S106 includes S110 and S111. S110 and S111 can be performed either in parallel or in series time-wise. In the case where they are performed in series time-wise, the order of S110 and S111 is arbitrary. In S110, of the process requestors included in the list information, a plurality of unselected ones are selected as the process requesters to be inserted and read, and the message is inserted into the message queues 39 related to the process requesters to be inserted and read. In S111, the agents related to the process requesters are read from the persistent storage 24 to the cache memory 25.

[0119] FIG. 12 is a flowchart of another message processing method. S100, S101, S104 and S105 have the same contents as those in FIG. 10 and a description thereof will be omitted. In S115 (selection step), either S116 (first message queue processing step) or S117 (second

message queue processing step) is selected. In S118 (agent instruction step), as to the agents 23 related to the message queue 39 having the message inserted therein, the agent 23 is immediately instructed to operate if the agent 23 is in the cache memory 25, and the agent 23 is read from the persistent storage 24 to the cache memory 25 and then instructed to operate if the agent 23 is not in the cache memory 25.

[0120] FIG. 13 shows the concrete process of S116 (first message queue processing step) in FIG. 12. S116 (first message queue processing step) includes S119 (insertion step) for inserting the message into the message queues 39 related to all the process requestors included in the list information.

[0121] FIG. 14 shows the concrete process of S117 (second message queue processing step) in FIG. 12. S117 includes S106 and S107. As S106 and S107 thereof are the same as S106 and S107 in FIG. 10, a description thereof will be omitted.

[0122] With reference to FIG. 12 again, the selection in S115 (selection step) is based on the instruction of the operator. Or the selection in S115 (selection step) is based on the estimated number of the process requestors to which the message generated due to the ~~agent start cause event~~ agent activating event of this time is applied, estimated hit rate in the cache memory 25 as to the agent 23 related to the process requesters to which the message generated due to the ~~agent start cause event~~ agent activating event of this time is applied, estimated work time, in the case where the process is allocated in S116 (the first message queue processing step), from the acceptance of the information in S104 (acceptance step) until obtaining the list information on the process requesters to which the message is applied and inserting the message into the message queues 39 of all the agents 23, estimated work time, in the case where the process is allocated in S117 (second message queue processing step), from the acceptance of the information in S104 (acceptance step) until obtaining the list information on the process requestors to which the message is applied and inserting the message into the message queues 39 of all the agents 23, estimated time from determination of use as to the agent 23 in the cache memory 25 until completion of the process by the determined agent 23,

and/or estimated time from determination of use as to the agent 23 outside the cache memory 25 until the completion of the process by the determined agent 23.

[0123] FIG. 15 is a flowchart of a further message processing method. S100, S101 and S104 are the same as those in FIG. 10, and a description thereof will be omitted. In S125 (process requestor determination step), the process requestor to which the message generated due to the ~~agent start cause event~~ agent activating event is applied is determined based on the process requestor search information 21. In at least one S128a, b, . . . (sub-process priority determination steps), the process priority about each message is determined as the sub-process priority based on the single standard of value. In S129 (compound process priority determination step), when the total number of the sub-process priority determination steps is two or more, the compound process priority about each message is determined based on the sub-process priority individually determined as to each message in each of the sub-process priority determination steps S128a, b, . . . , and when the total number of the sub-process priority determination steps is one, the sub-process priority determined in the one sub-process priority determination step (ex. S128a) as to each message is determined as the compound process priority. In S132 (agent instruction step), the message of the highest compound process priority among the messages held by each message queue is rendered as the message of the highest priority, and between the agents 23 related to the message queues of which compound process priority of the message of the highest priority is the same, the agent 23 existing in the cache memory 25 is instructed to operate in preference to the agent 23 not existing therein.

[0124] The standards of value adopted in S128a, S128b, . . . (sub-process priority determination steps) are related either to the contents of the message or to the process requestors to which the message is applied. Furthermore, the predetermined standard of value related to the contents of the message includes the one related to the emergency of processing of the message. The standard of value related to the process requesters to which the message is applied includes the one related to the rating of the process requestors.

[0125] When there are a plurality of messages held by the message queue related to the agent 23 having started the process in S132 (agent instruction step), the agent 23 continuously processes all those messages or the messages of which compound process priority is within the predetermined position in descending rank.

[0126] FIG. 16 is a flowchart of a portion of the message processing method having a high-order step including S128a, S128b, . . . and S129 in FIG. 15 as sub-steps. S135 (agent management step) includes S137 and S138 in addition to S128a, 128b, . . . and S129. In S137 (existence detection step), it is detected whether or not each agent 23 exists in the persistent storage 24. In S138 (grouping information management step), the agents 23 are grouped, grouping information is managed, and the grouping information is updated as appropriate based on determination results of the S137 (existence detection step) and the compound process priority of each agent 23. In S132 (agent instruction step), the agents 23 are instructed to operate in order based on the grouping information in the agent management step.

[0127] FIG. 17 is a flowchart of a still further message processing method. S100, S101 and S104 are the same as those in FIG. 10, and a description thereof will be omitted. In S149 (acceptance order information management step), the acceptance order information on the ~~agent start cause events~~ agent activating events accepted in S104 (acceptance step) is managed. In S150 (thread setting step), a plurality of threads 67a, 67b, . . . are set up. According to this setup, the threads 67a, 67b, . . . are mutually operable in parallel, and detect the process requestors to which the message generated due to the ~~agent start cause event~~ agent activating event is applied based on the process requestor search information 21 and insert the message into the message queues 39 related to the process requesters. In S152 (allocation step), the ~~agent start cause event~~ agent activating event to be processed by each of the threads 67a, 67b, . . . is allocated thereto. Each thread thus having it allocated starts the operation in S153. In S153 (proceeding information management step), the proceeding information on the process by the threads 67a, 67b, . . . as to each ~~agent start cause event~~ agent activating event accepted in S104 (acceptance step) is managed. In S156 (determination step), as to the ~~agent start cause event~~ agent activating event of which process proceeding information is the

information on process termination by the threads 67a, 67b, . . . (hereafter, referred to as a "determined ~~agent start cause event~~ agent activating event"), it is determined whether or not, of the ~~agent start cause events~~ agent activating events accepted in S104 (acceptance step) prior to the determined ~~agent start cause event~~ agent activating event, there is any ~~agent start cause event~~ agent activating event of which process by the threads 67a, 67b, . . . is unfinished.

[0128] FIG. 18 shows a concrete example of delivery process control of S157 (agent control step) in FIG. 17. In S160, it branches to S161 or S162 based on the determination result of S156. To be more specific, it proceeds to S161 in the case where the determination result of S156 is "yes," and it proceeds to S162 in the case where it is "no."

[0129] In S161, the process by the agent 23 is controlled as to the message generated due to the determined ~~agent start cause event~~ agent activating event determined as "yes" in S156 (determination step). In S162, the process by the agent 23 is allowed as to the message generated due to the determined ~~agent start cause event~~ agent activating event determined as "no" in S156 (determination step).

[0130] In the determination step S156, it is preferable that, in the case where the ~~agent start cause event~~ agent activating event immediately following the ~~agent start cause event~~ agent activating event determined as "no" in the acceptance order is already determined as "yes," the determination result is changed from "yes" to "no." Thus, the process by the agent 23 is promptly started as to the message generated due to the next ~~agent start cause event~~ agent activating event. Furthermore, as for the setup of the agent 23 in S101 (agent setting step), it is preferable that, in the case of causing the agent 23 to process the message queue 39 in which the messages generated due to a plurality of ~~agent start cause events~~ agent activating events of which determination result by S156 (determination step) is "no" continue in the acceptance order direction, the agent 23 is set to continuously process the plurality of continuous messages.

[0180] According to this method, an adequate number of agent keys is read from the list of the agent keys and the message is put in the message queue thereof, and if the agent completes the process of the message, the next adequate number of agent keys are obtained from the list so as to perform the same process. In this case, as the SwapInCollection is utilized, the agents corresponding to the keys which are not in the agent cache are collectively swapped in concurrently with reading of the agent keys from the list. It is possible, by this reading, to swap in a plurality of agents at a very high speed compared to the case where ~~one SQL search for the sake of~~ the Swap In of one agent is repeatedly performed a plurality of times. Therefore, it is effective in the case where the agent cache size is smaller than that of the list of the agent keys and the cache hit rate is low.

[0181] The start of each agent is managed by the scheduler as a mechanism separate from the KeyOnlyCollection delivery mechanism and SwapInCollection delivery mechanism. If the agent to be started is not in the agent cache, the scheduler reads the agent from the persistent area to the agent cache. When the SwapInCollection delivery mechanism is in operation, it is ensured that the agent related to the message queue having the delivery message inserted therein is read into the cache memory so that the scheduler does not need to read the agent ~~to be started~~ from the persistent area to the agent cache ~~saved~~.

[0197] (5) The estimated value of the time required in the KeyOnly delivery mechanism is measured, after receiving the message, from ~~obtain~~ obtaining the list of key information on the agents to which it should be delivered and ~~put~~ to putting the message in the message queues of all the agents corresponding to the obtained key information (hereafter, referred to as "T.sub.k"). It does not include the time for the agents to process the message.

[0229] FIG. 21 shows a data structure of the control block. A first subject table 230 has ~~correspondence of~~ the agent key (AgentKey) ~~to~~ and the control block (ControlBlock) which ~~is~~ are 1:1 recorded therein. The agent key is intended to identify the agent, and corresponds to each delivery destination at 1:1. In the first subject table 230 in FIG. 21, the agent keys are represented by personal names (Mike, Tom and Eric). A second subject table 231 has the

correspondence of the agent type (AgentType) to the priority (Priority) recorded therein. In this example, there are two agent types of Gold and Normal, and the priorities 1 and 0 are set to the Gold and Normal respectively. As previously mentioned, it is also possible to widen the difference by setting the priorities of the gold members and normal members at 5 and 0 respectively, for instance. Each control block 232 includes the data on the message queue associated with it (MessageQueue queue), data on the preceding control block 232 linked to it (ControlBlock prev), data on the subsequent control block 232 linked to it (ControlBlock next), data representing the current state of it (byte state, such as IN_FILLED and IN_EMPTY mentioned later) and data on a string type (String type).

[0332] Thus, according to the present invention, it is possible, by determining the process requestors to which the message generated due to the accepted ~~agent start cause event~~ agent activating event is applied based on the process requestor search information, to effectively control the reading of the agent associated with each process requestor from the persistent storage to the cache memory so as to realize the reduction in the processing time.